CSEE 3827: Fundamentals of Computer Systems

Boolean Algebra

M&K 2.3-2.5

Agenda

- Standard Forms
  - Product-of-Sums (PoS)
  - Sum-of-Products (SoP)
    - conversion between
  - Min-terms and Max-terms
- Simplification via Karnaugh Maps (K-maps)
  - 2, 3, and 4 variable
  - Implicants, Prime Implicants, Essential Prime Implicants
  - Using K-maps to reduce
  - PoS form
  - Don’t Care Conditions

Standard Forms

- There are many ways to express a boolean expression

$$F = XYZ + XYZ + XZ$$
$$= XY(Z + Z) + XZ$$
$$= XY + XZ$$

- It is useful to have a standard or canonical way
- Derived from truth table
- Generally not the simplest form

Two principle standard forms

- Sum-of-products (SOP)

- Product-of-sums (POS)

Product and sum terms

- Product term: logical AND of literals (e.g., $X\overline{Y}Z$)
- Sum term: logical OR of literals (e.g., $A + \overline{B} + C$)

PoS & SoP

- Sum of products (SoP): OR of ANDs

$$\text{e.g., } F = \overline{Y} + \overline{X}Y\overline{Z} + XY$$

- Product of sums (PoS): AND of ORs

$$\text{e.g., } G = X(\overline{Y} + Z)(X + Y + \overline{Z})$$

Converting from PoS (or any form) to SoP

Just multiply through and simplify, e.g.,

$$
\begin{aligned}
G &= X(Y + Z)(X + Y + Z) \\
&= XYX + XYY + XYZ + XZX + XZY + XZZ \\
&= XY + XY + XYZ + XZ + XZY + XZ \\
&= XY + XZ
\end{aligned}
$$

Converting from SoP to PoS

Complement, multiply through, complement via DeMorgan, e.g.,

Note: $X' = \overline{X}$

$$F = Y'Z' + XY'Z + XYZ'$$

$$F' = (Y+Z)(X'+Y+Z')(X'+Y'+Z)$$

$$= YZ + X'Y + X'Z$$ (after lots of simplification)

$$F = (Y'+Z')(X+Y')(X+Z')$$

Minterms

e.g., Minterms for 3 variables A,B,C

A	B	C	minterm
0	0	0	m0 $\bar{A}\bar{B}\bar{C}$
0	0	1	m1 $\bar{A}\bar{B}C$
0	1	0	m2 $\bar{A}B\bar{C}$
0	1	1	m3 $\bar{A}BC$
1	0	0	m4 $A\bar{B}\bar{C}$
1	0	1	m5 $A\bar{B}C$
1	1	0	m6 $AB\bar{C}$
1	1	1	m7 ABC

- A product term in which all variables appear once, either complemented or uncomplemented (i.e., an entry in the truth table).
- Each minterm evaluates to 1 for exactly one variable assignment, 0 for all others.
- Denoted by mX where X corresponds to the variable assignment for which mX = 1.

Minterms to describe a function

sometimes also called a ***minterm expansion*** *or* ***disjunctive normal form (DNF)***

This “term” is TRUE when
A=0,B=1,C=0

$$F = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C$$

A	B	C	F	$\overline{F}$
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	0	1
1	1	1	0	1

$$\overline{F} = \overline{A}BC + AB\overline{C} + ABC$$

Minterm example, seen another way

The logical OR of all minterms for which F = 1.

A	B	C	minterm		F
0	0	0	m0	$\bar{A}\bar{B}\bar{C}$	1
0	0	1	m1	$\bar{A}\bar{B}C$	1
0	1	0	m2	$\bar{A}B\bar{C}$	1
0	1	1	m3	$\bar{A}BC$	0
1	0	0	m4	$A\bar{B}\bar{C}$	1
1	0	1	m5	$A\bar{B}C$	1
1	1	0	m6	$AB\bar{C}$	0
1	1	1	m7	ABC	0

m0		m1		m2	m3	m4		m5	m6	m7
1	+	0	+	0	0	0	+	0	0	0
0	+	1	+	0	0	0	+	0	0	0
0	+	0	+	1	0	0	+	0	0	0
0	+	0	+	0	1	0	+	0	0	0
0	+	0	+	0	0	1	+	0	0	0
0	+	0	+	0	0	0	+	1	0	0
0	+	0	+	0	0	0	+	0	1	0
0	+	0	+	0	0	0	+	0	0	1

Minterm example, conclusion

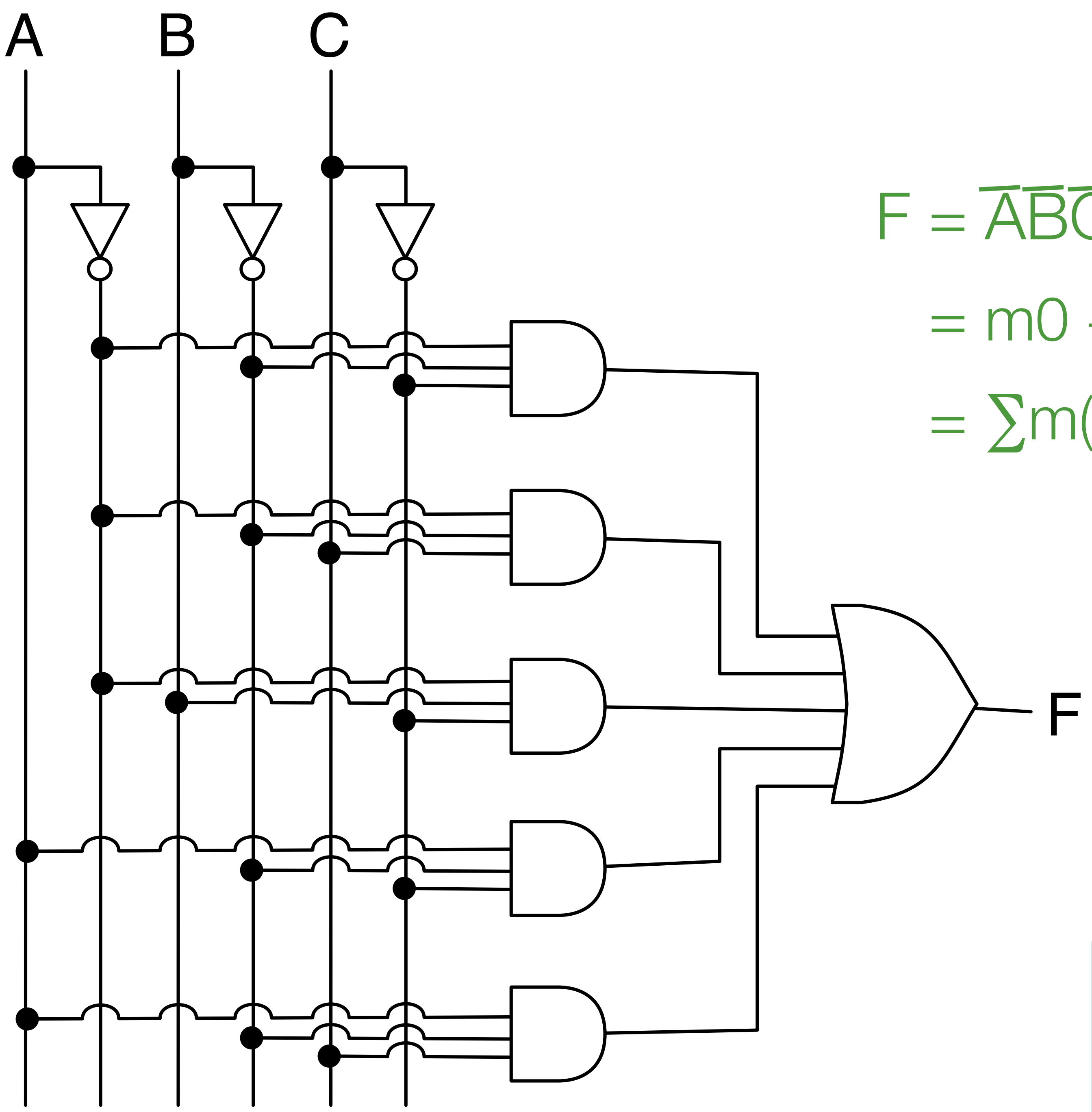
(variables appear once in each minterm)

A	B	C	F	$\overline{F}$	minterm
0	0	0	1	0	m0 $\overline{A}\,\overline{B}\,\overline{C}$
0	0	1	1	0	m1 $\overline{A}\,\overline{B}C$
0	1	0	1	0	m2 $\overline{A}B\overline{C}$
0	1	1	0	1	m3 $\overline{A}BC$
1	0	0	1	0	m4 $A\overline{B}\,\overline{C}$
1	0	1	1	0	m5 $A\overline{B}C$
1	1	0	0	1	m6 $AB\overline{C}$
1	1	1	0	1	m7 ABC

$$F = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C$$
$$= m0 + m1 + m2 + m4 + m5$$
$$= \sum m(0,1,2,4,5)$$

$$\overline{F} = \overline{A}BC + AB\overline{C} + ABC$$
$$= m3 + m6 + m7$$
$$= \sum m(3,6,7)$$

Minterms as a circuit



$$F = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C$$
$$= m0 + m1 + m2 + m4 + m5$$
$$= \sum m(0,1,2,4,5)$$

Standard form is not minimal form!

Maxterms

A	B	C	maxterm	
0	0	0	M0	$A+B+C$
0	0	1	M1	$A+B+\overline{C}$
0	1	0	M2	$A+\overline{B}+C$
0	1	1	M3	$A+\overline{B}+\overline{C}$
1	0	0	M4	$\overline{A}+B+C$
1	0	1	M5	$\overline{A}+B+\overline{C}$
1	1	0	M6	$\overline{A}+\overline{B}+C$
1	1	1	M7	$\overline{A}+\overline{B}+\overline{C}$

- A sum term in which all variables appear once, either complemented or uncomplemented.
- Each maxterm evaluates to 0 for exactly one variable assignment, 1 for all others.
- Denoted by MX where X corresponds to the variable assignment for which MX = 0.

Maxterm description of a function

sometimes also called a ***maxterm expansion*** *or* ***conjunctive normal form (CNF)***

This "term" is FALSE when
A=1,B=1,C=0

$$F = (A+\overline{B}+\overline{C})\,(\overline{A}+\overline{B}+C)\,(\overline{A}+\overline{B}+\overline{C})$$

A	B	C	F	$\overline{F}$
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	0	1
1	1	1	0	1

Force to 0

Maxterm example, seen another way

The logical AND of all maxterms for which F = 0.

A	B	C	maxterm		F
0	0	0	M0	$A+B+C$	1
0	0	1	M1	$A+B+\overline{C}$	1
0	1	0	M2	$A+\overline{B}+C$	1
0	1	1	M3	$A+\overline{B}+\overline{C}$	0
1	0	0	M4	$\overline{A}+B+C$	1
1	0	1	M5	$\overline{A}+B+\overline{C}$	1
1	1	0	M6	$\overline{A}+\overline{B}+C$	0
1	1	1	M7	$\overline{A}+\overline{B}+\overline{C}$	0

M0	M1	M2	M3	M4	M5	M6	M7
0	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1
1	1	0	1	1	1	1	1
1	1	1	0	1	1	1	1
1	1	1	1	0	1	1	1
1	1	1	1	1	0	1	1
1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	0

Maxterm example, conclusion

The logical AND of all maxterms for which F = 0.

A	B	C	maxterm		F
0	0	0	M0	$A+B+C$	1
0	0	1	M1	$A+B+\overline{C}$	1
0	1	0	M2	$A+\overline{B}+C$	1
0	1	1	M3	$A+\overline{B}+\overline{C}$	0
1	0	0	M4	$\overline{A}+B+C$	1
1	0	1	M5	$\overline{A}+B+\overline{C}$	1
1	1	0	M6	$\overline{A}+\overline{B}+C$	0
1	1	1	M7	$\overline{A}+\overline{B}+\overline{C}$	0

$$F = (A+\overline{B}+\overline{C})\,(\overline{A}+\overline{B}+C)\,(\overline{A}+\overline{B}+\overline{C})$$

$$= (M0)\,(M4)\,(M5)\,(M6)\,(M7)$$

$$= \prod M(0,4,5,6,7)$$

One final example

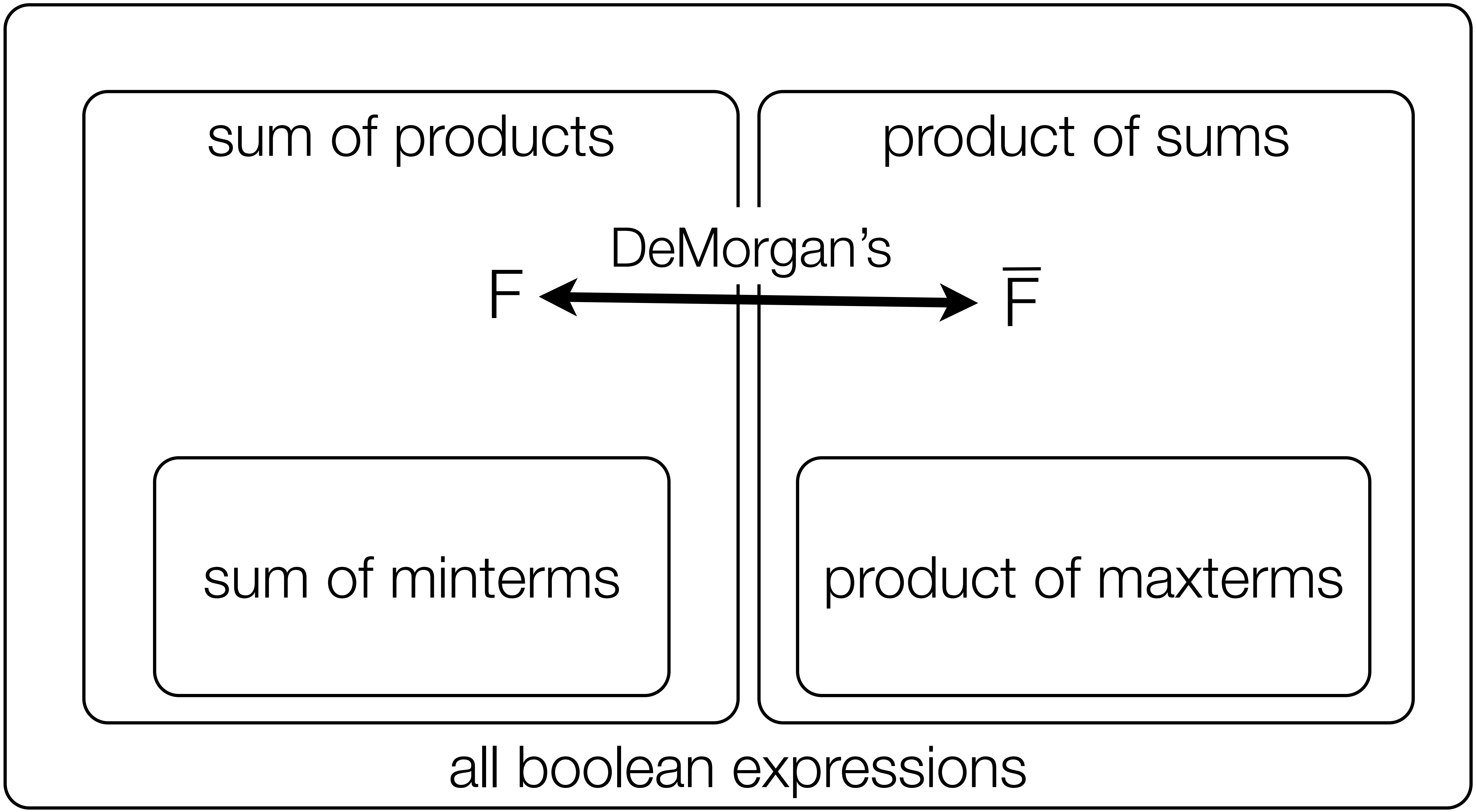
A	B	C	F	$\overline{F}$
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	0	1

	F	$\overline{F}$
Minterms (SOP)		
Maxterms (POS)		

Summary of Minterms and Maxterms

	F	$\overline{F}$
Minterms (SOP)	$\sum m(F = 1)$	$\sum m(F = 0)$
Maxterms (POS)	$\prod M(F = 0)$	$\prod M(F = 1)$

Relations between standard forms



Simplification with Karnaugh Maps

Cost criteria

- Literal cost: the number of literals in an expression
- Gate-input cost: the literal cost + all terms with more than one literal + (optionally) the number of distinct, complemented single literals

Roughly proportional to the number of transistors and wires in an AND/OR/NOT circuits. Does not apply, to more complex gates, for example XOR.

	Literal cost	Gate-input cost
$G = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + ABCD$	8	8 + 2 + (4)
$G = (\overline{A}+B)(\overline{B}+C)(\overline{C}+D)(\overline{D}+A)$	8	8 + 5 + (4)

Karnaugh maps (a.k.a., K-maps)

- All functions can be expressed with a map

- There is one square in the map for each minterm in a function’s truth table

X	Y	F
0	0	m0
0	1	m1
1	0	m2
1	1	m3

X \ Y	0	1
0	m0 $\overline{X}\,\overline{Y}$	m1 $\overline{X}Y$
1	m2 $X\overline{Y}$	m3 XY

Karnaugh maps express functions

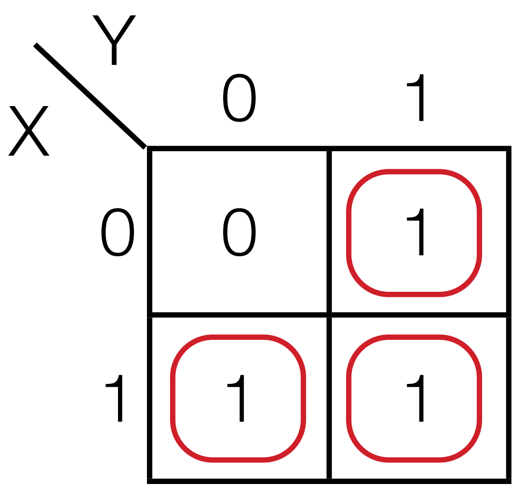
- Fill out table with value of a function

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	1

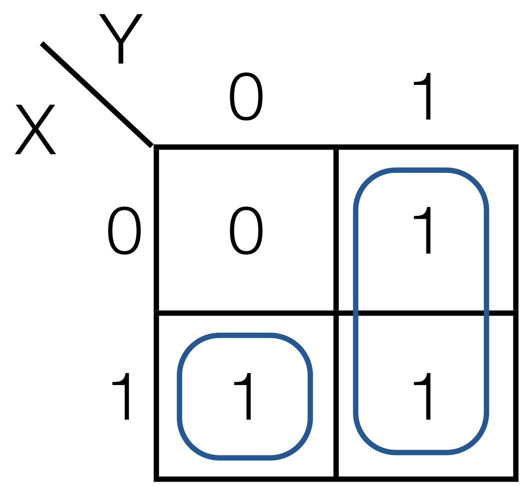
X \ Y	0	1
0		
1		

Simplification using a k-map

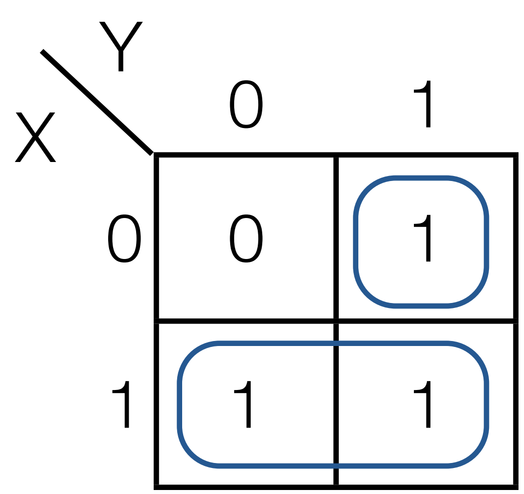
- Whenever two squares share an edge and both are 1, those two terms can be combined to form a single term with one less variable



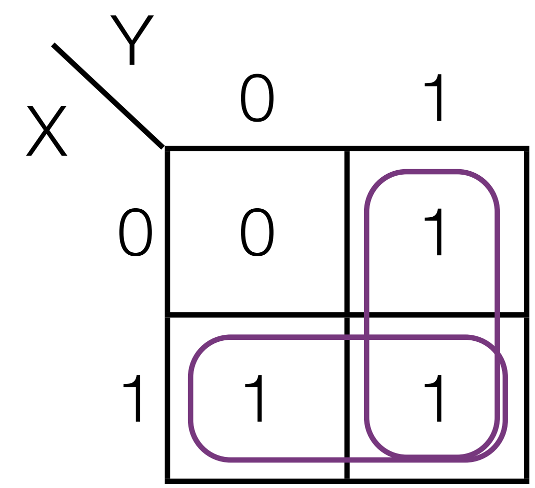
$F = \overline{X}Y + X\overline{Y} + XY$



$F = Y + X\overline{Y}$



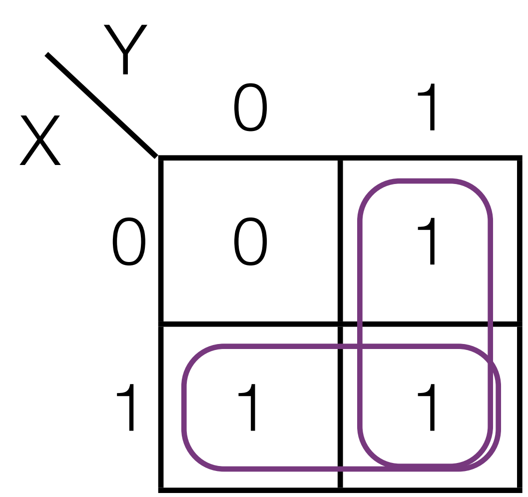
$F = X + \overline{X}Y$



$F = X + Y$

Simplification using a k-map (2)

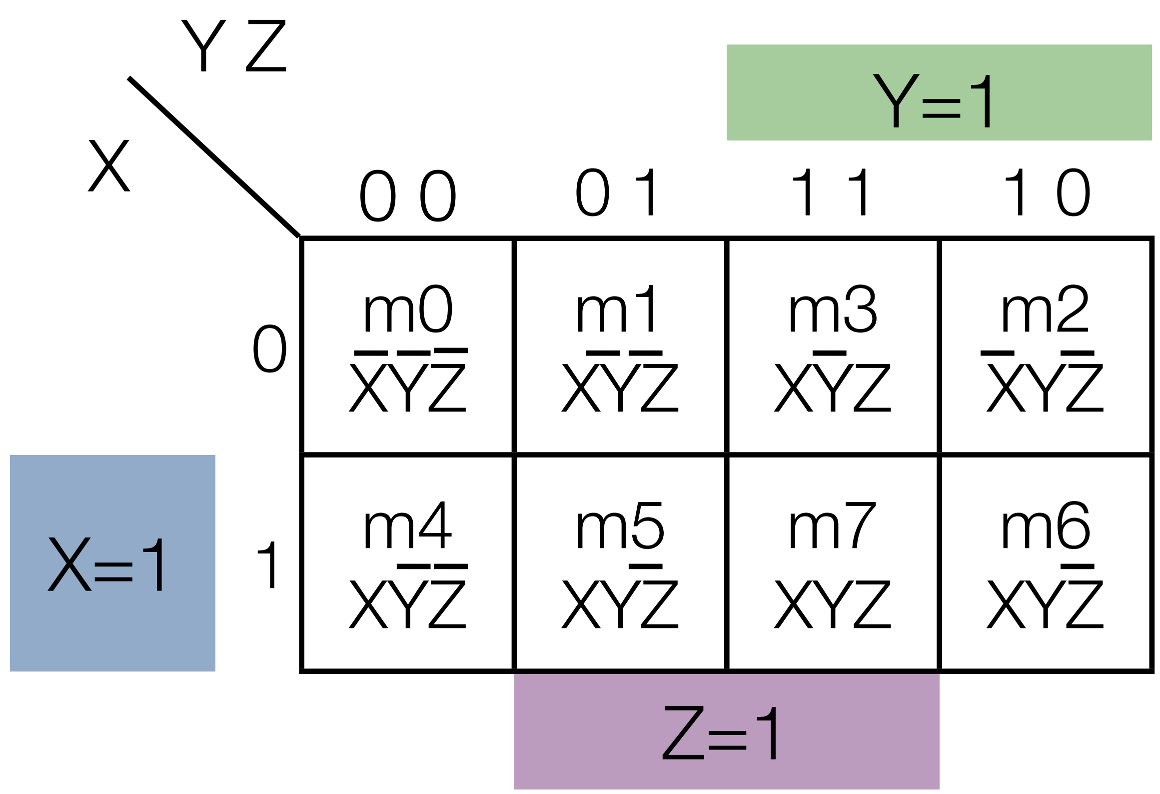
- Circle contiguous groups of 1s (circle sizes must be a power of 2)
- There is a correspondence between circles on a k-map and terms in a function expression
- The bigger the circle, the simpler the term
- Add circles (and terms) until all 1s on the k-map are circled



$$F = X + Y$$

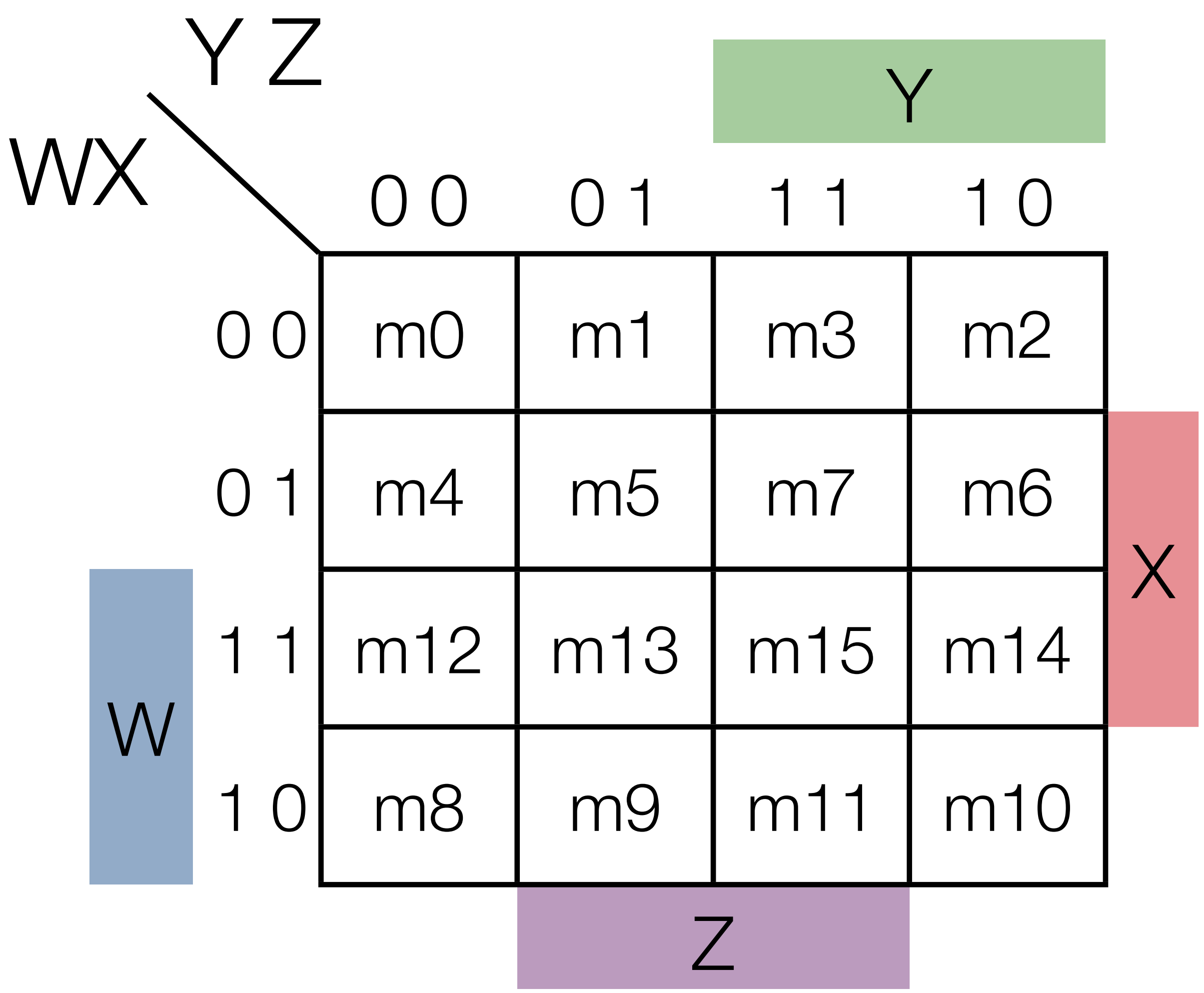
3-variable Karnaugh maps

- Use gray ordering on edges with multiple variables
- Gray encoding: order of values such that only one bit changes at a time
- Two minterms are considered adjacent if they differ in only one variable (this means maps “wrap”)



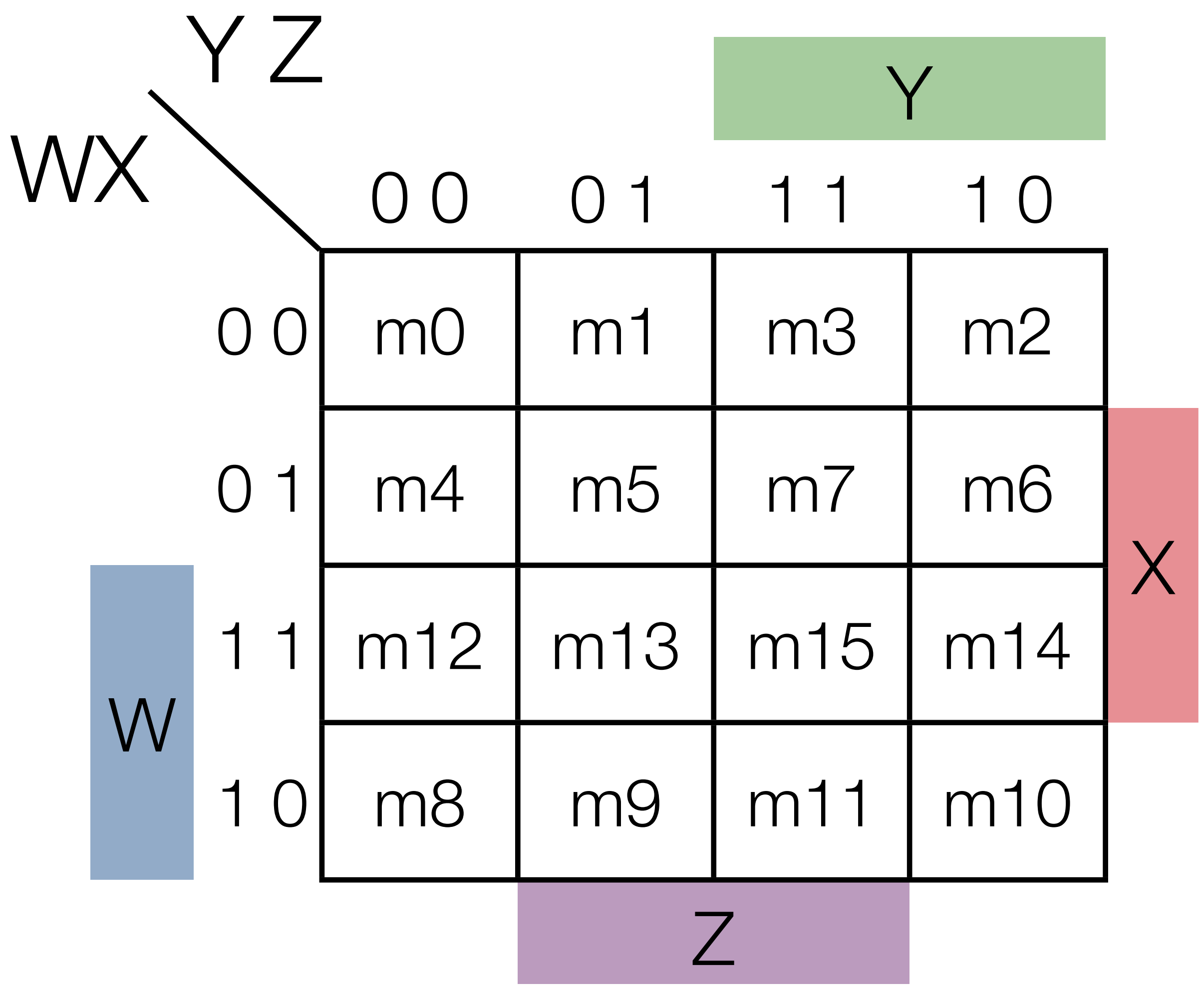
4-variable Karnaugh maps

Extension of 3-variable maps



Implicants

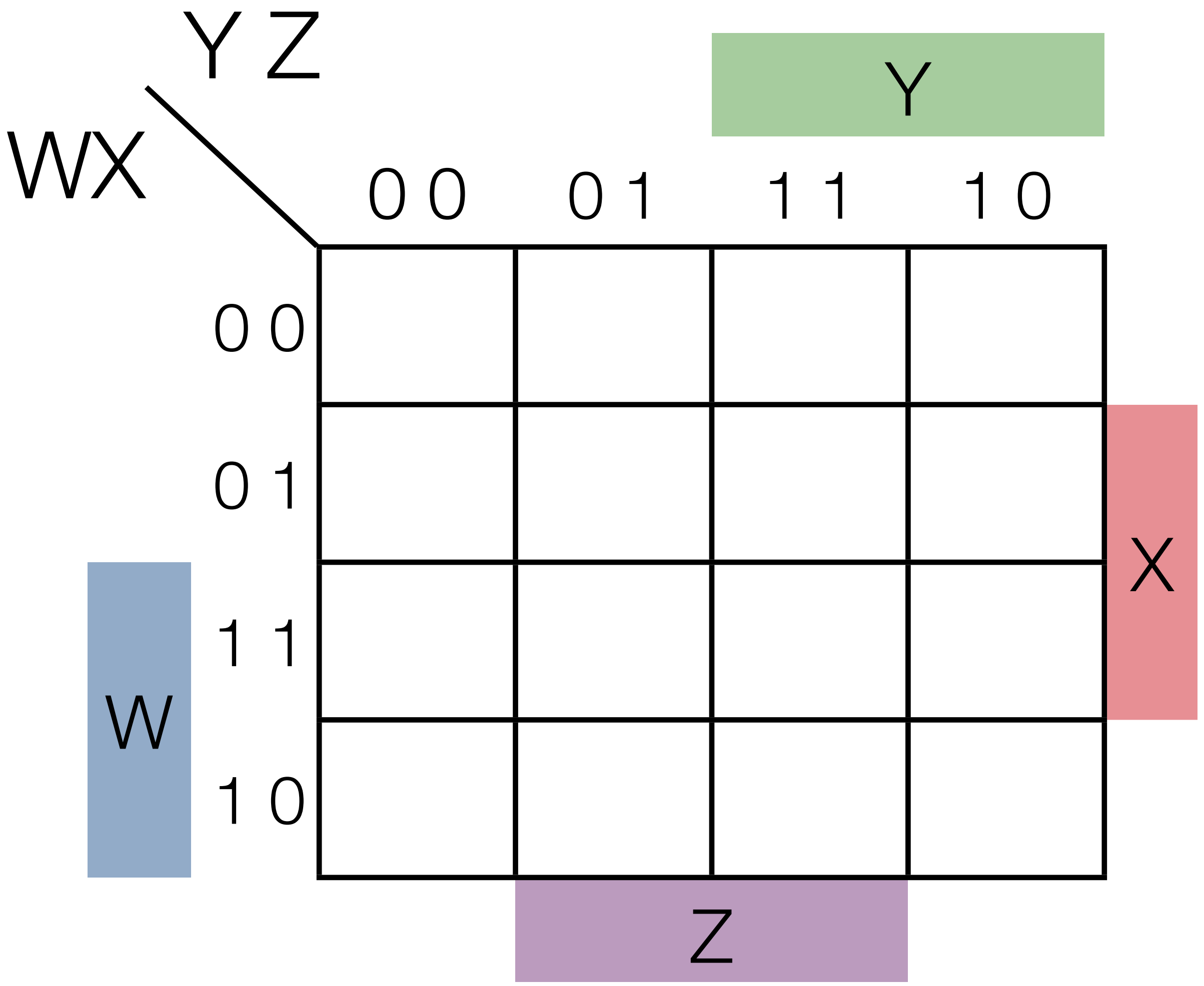
***Implicant**: a product term, which, viewed in a K-Map is a 2^i x 2^j size “rectangle” (possibly wrapping around) where i=0,1,2, j=0,1,2*



Note: bigger rectangles = fewer literals

4-variable Karnaugh map example

W	**X**	**Y**	**Z**	**F**
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0



Implicant terminology

- **implicant**: a product term, which, viewed in a K-Map is a 2^i x 2^j size "rectangle" (possibly wrapping around) where i=0,1,2, j=0,1,2
- **prime implicant**: An implicant not contained within another implicant.
- **essential prime implicant**: a **prime implicant** that is the **only prime implicant** to cover some minterm.

4-variable Karnaugh maps (3)

- List all of the prime implicants for this function
- Is any of them an essential prime implicant?
- What is a simplified expression for this function?

WX \ YZ	00	01	11	10
00	0	0	1	0
01	1	1	1	0
11	0	1	1	1
10	0	1	0	0

Using K-maps to build simplified circuits

- Step 1: Identify all PIs and essential PIs
- Step 2: Include all Essential PIs in the circuit (Why?)
- Step 3: If any 1-valued minterms are uncovered by EPIs, choose PIs that are “big” and do a good job covering

1	1	1	0
0	1	1	0
1	1	1	1
1	1	0	1

1	1	0	0
0	1	1	0
0	0	1	1
1	0	0	1

Design example : 2-bit multiplier

a_1	a_0	b_1	b_0	z_3	z_2	z_1	z_0
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

two 2-bit #'s multiplied together to give a 4-bit solution

e.g., $a_1a_0 = 10$, $b_1b_0 = 11$, $z_3z_2z_1z_0 = 0110$

K-Maps: Complements, PoS, don’t care conditions

Finding $\overline{F}$

Find prime implicants corresponding to the 0s on a k-map

WX \ YZ	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	0	0
10	1	1	0	1

WX \ YZ	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	0	0	1	1
10	0	0	1	0

$$F = \overline{Y} + \overline{X}\,\overline{Z} + \overline{W}\,\overline{Z}$$

$$\overline{F} = YZ + WXY$$

PoS expressions from a k-map

Find $\overline{F}$ as SoP and then apply DeMorgan's

WX \ YZ	0 0	0 1	1 1	1 0
0 0	1	1	0	1
0 1	1	0	0	0
1 1	1	0	0	0
1 0	1	1	0	1

$$\overline{F} = YZ + XZ + YX$$

DeMorgan's

$$F = (\overline{Y}+\overline{Z})(\overline{Z}+\overline{X})(\overline{Y}+\overline{X})$$

Don’t care conditions

There are circumstances in which the value of an output doesn’t matter

- For example, in that 2-bit multiplier, what if we are told that a and b will be non-0? We “don’t care” what the output looks like for the input cases that should not occur

- Don’t care situations are denoted by an “X” in a truth table and in Karnaugh maps.

- Can also be expressed in minterm form:

- During minimization can be treated as either a 1 or a 0

$$z2 = \sum m(10,11,14)$$
$$d2 = \sum m(0,1,2,3,4,8,12)$$

a1	a0	b1	b0	z3	z2	z1	z0
0	0	0	0	X	X	X	X
0	0	0	1	X	X	X	X
0	0	1	0	X	X	X	X
0	0	1	1	X	X	X	X
0	1	0	0	X	X	X	X
0	1	0	1	0	0	0	1
0	1	1	0	0	0	1	0
0	1	1	1	0	0	1	1
1	0	0	0	X	X	X	X
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	X	X	X	X
1	1	0	1	0	0	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	0	0	1

2-bit multiplier non-0 multiplier

a1	a0	b1	b0	z3	z2	z1	z0
0	0	0	0	X	X	X	X
0	0	0	1	X	X	X	X
0	0	1	0	X	X	X	X
0	0	1	1	X	X	X	X
0	1	0	0	X	X	X	X
0	1	0	1	0	0	0	1
0	1	1	0	0	0	1	0
0	1	1	1	0	0	1	1
1	0	0	0	X	X	X	X
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	X	X	X	X
1	1	0	1	0	0	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	0	0	1

1's must be covered
0's must not be covered
X's are optionally covered

b_0

X	X	X	X
X	0	0	0
X	0	1	0
X	0	0	0

a_0 a_1 b_1

$z_3 =$

b_0

X	X	X	X
X	0	0	0
X	0	0	1
X	0	1	1

a_0 a_1 b_1

$z_2 =$

2-bit multiplier non-0 multiplier (2)

a1	a0	b1	b0	z3	z2	z1	z0
0	0	0	0	X	X	X	X
0	0	0	1	X	X	X	X
0	0	1	0	X	X	X	X
0	0	1	1	X	X	X	X
0	1	0	0	X	X	X	X
0	1	0	1	0	0	0	1
0	1	1	0	0	0	1	0
0	1	1	1	0	0	1	1
1	0	0	0	X	X	X	X
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	X	X	X	X
1	1	0	1	0	0	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	0	0	1

b_0

X	X	X	X
X	0	1	1
X	1	0	1
X	1	1	0

a_0 a_1 b_1

b_0

X	X	X	X
X	1	1	0
X	1	1	0
X	0	0	0

a_0 a_1 b_1

$z_1 =$

$z_0 =$

Final thoughts on Don’t care conditions

Sometimes “don’t cares” greatly simplify circuitry

D

	1	X	X	X
	X	1	X	X
	0	0	1	X
A	0	0	X	1

B

C

$$\overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}B\overline{C}D + ABCD + A\overline{B}C\overline{D} \quad \text{vs. } \overline{A} + C$$

Glitches and Hazards

Glitches and hazards

- Glitch: an unintended change in circuit output
- Hazard: the hardware structures that cause a glitch to occur
- Caused by multiple path delays through a circuit
- Example: $\bar{A}\bar{B} + BC$
- Avoidance
  - Synchronous design (coming later)
  - Extra implicants